

รายงานวิชาปฏิบัติการ

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

รหัสวิชา ____242-301____ ตอน ____01____ วัน ____อังคาร____

รหัสหัวข้อปฏิบัติการ ____2HA06____

ชื่อหัวข้อปฏิบัติการ ____Creating sound using FPGA____

วันที่ลงปฏิบัติการ ____24 ตุลาคม 2560____

อาจารย์ผู้สอน ____อาจารย์สุธาสนิย์ เสน่ห์เสาวรส____

กลุ่มที่ ____8____

ผู้จัดทำรายงานชื่อ ____นายปณิธาน ดวงขวัญ____ รหัส ____5735512036____

ผู้ร่วมงาน ชื่อ ____นางสาวรัตนพร ไทยเกิด____ รหัส ____5735512047____

ชื่อ ____รหัส ____

สำหรับเจ้าหน้าที่

วันที่ตรวจรับ ____

ลงชื่อ ____



การทดลองที่ 3HA06

Creating sound using FPGA

เครื่องมือและอุปกรณ์

1. โปรแกรม Xilinx
2. บอร์ด FPGA รุ่น XC3S200-TQ144C
3. คอมพิวเตอร์

การทดลองที่ 1

แนะนำการสร้างวงจร Buzzer อย่างง่ายโดยใช้ switch เป็นตัว on-off และขับลอจิก “1” ไปที่ Buzzer (speaker = 1)

```
1 `timescale 1ns / 1ps
2 //////////////////////////////////////
3 // Company:
4 // Engineer:
5 //
6 // Create Date:    09:55:53 10/24/2017
7 // Design Name:
8 // Module Name:    buzzer1
9 // Project Name:
10 // Target Devices:
11 // Tool versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////
21 module buzzer1(
22     input clk,
23     input onoff,
24     output reg sp
25 );
26
27     always@(onoff)
28     if(onoff == 1)
29         sp <= 1;
30     else
31         sp <= 0;
32
33     //assign sp = (onoff ? 1:0);
34
35 endmodule
36
37
```

สรุปผลการทดลอง

จากโค้ดในรูปภาพประกอบแล้วโดยจะสั่งให้บอร์ด FPGA นั้นส่งเสียงสัญญาณโดยขึ้นกับขาที่เรากำหนดจากโปรแกรมโดยสามารถเปิดและปิดเสียงได้ตามที่เราต้องการโดยสับสวิตช์บนบอร์ด FPGA



การทดลองที่ 2

2 ต่อเนื่องจาก Code การทดลองที่ 1 ให้สร้างวงจรความถี่จาก 25MHz(clock) ให้เหลือ 440 Hz สังเกตเสียงระหว่างการทดลองที่ 1 และ 2

```
1 `timescale 1ns / 1ps
2 //////////////////////////////////////////////////
3 // Company:
4 // Engineer:
5 //
6 // Create Date:    10:07:34 10/24/2017
7 // Design Name:
8 // Module Name:    buzzer2
9 // Project Name:
10 // Target Devices:
11 // Tool versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////
21 module buzzer2(
22     input clk,
23     input onoff,
24     output reg |sp
25 );
26
27 parameter clk440 = ((25000000/440)/2);
28
29 reg [31:0] cnt;
30
31 always@(posedge clk)
32     if(onoff == 0)
33         sp <= 0;
34     else begin
35         if(cnt == clk440) begin sp<= ~sp; cnt <=0; end
36         else begin cnt <= cnt + 1; end
37     end
38
39 endmodule
```

สรุปผลการทดลอง

จากภาพประกอบด้านบนนั้นเป็นการเขียนโปรแกรมเพื่อให้บอร์ด FPGA นั้นส่งเสียงความถี่ ความถี่ที่เราต้องการโดยใช้สัญญาณนาฬิกาเข้าช่วยในการขับเคลื่อนความถี่โดยเราใช้คลื่นความถี่เป็น 440Hz โดยเราจะได้ยินเสียง buzzer จากบอร์ดทดลองเป็นเสียงคลื่นเล็กๆแหลมๆ



การทดลองที่ 3

ต่อเนื่องจาก Code การทดลองที่ 2 สร้างวงจร Siren “ป๊อป” ใช้ 2 ความถี่จากเดิม 440 Hz และ 220 Hz

```
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////
21 module buzzer3(
22     input clk,
23     input onoff,
24     output reg sp
25 );
26
27 parameter clk440= (25000000/440/2);
28 parameter clk220= (25000000/220/2);
29
30 reg speed;
31 reg sp1;
32 reg sp2;
33
34 always@(onoff)
35     if(onoff == 0) sp<=0;
36     else sp <= (speed ? sp1:sp2);
37
38 reg [31:0] cnt1;
39 always@(posedge clk)
40     if (cnt1 == clk440) begin sp1 <= ~sp1; cnt1 <= 0; end
41     else begin cnt1 <= cnt1 + 1; end
42
43 reg [31:0] cnt2;
44 always@(posedge clk)
45     if (cnt2 == clk220) begin sp2 <= ~sp2; cnt2 <= 0; end
46     else begin cnt2 <= cnt2 + 1; end
47
48 reg [31:0] cnt;
49 always@(posedge clk)
50     if(cnt == 25000000)
51         begin speed = ~speed; cnt <= 0; end
52     else cnt <= cnt + 1;
53
54 endmodule
55
```

สรุปผลการทดลอง

จากการทดลองนั้นโดยเราจะทำการเพิ่มโค้ดโดยใช้คลื่นความถี่สองชุดในการทำงานโดยความถี่ทั้งสองชุดนั้นจะมีค่าความถี่ที่แตกต่างกันโดยความถี่แรกจะเท่ากับ 440 และความถี่ที่สองจะเท่ากับ 220 โดยหลังจากนั้นนำทั้งสองคลื่นความถี่นั้นมาเข้าสัญญาณนาฬิกาเพื่อทำให้เกิดการสลับกันของคลื่นความถี่นั่นเอง โดยทั้งสองความถี่นั้นจะเสียงที่แตกต่างกันนั่นเองโดยจะดัง ป๊อป ตามโจทย์ที่อาจารย์ให้มา



การทดลองที่ 4

สร้างวงจรนับ 2 หลัก แสดงบน 7 Segment (หลักสิบ(ten) และ หลักหน่วย (one)) ให้ถูกต้อง

```

21 module buzzer1(
22   input clk,
23   input en,
24   output reg sp,
25   output [10:0] s1,
26   output [10:0] s0m
27 );
28   parameter M0 = 0;
29   parameter M1 = 1;
30
31
32   reg [10:0] s0m1;
33   reg [10:0] s0m0;
34   reg [10:0] s01;
35   reg [10:0] s02;
36   reg [10:0] s03;
37   reg [10:0] s0m_s0m0;
38   reg s0m1_s0m0;
39   reg s0m1_s01;
40   reg s0m1_s02;
41   reg s0m1_s03;
42   assign s0 = s0;
43   assign s0m = s0m0;
44
45   always@posedge clk1
46   if (en == 0) begin one <= 0; ten <= 0; end
47   else
48     begin
49       if (one == 9) begin one <= 0; ten <= ten + 1; end else begin one <= one + 1; end
50       if ((ten == 9) || (ten == 9)) begin ten <= 0; end
51     end
52
53   always@clk0
54   always@clk1
55   always@clk2
56   always@clk3
57   always@clk4
58   always@clk5
59   always@clk6
60   always@clk7
61   always@clk8
62   always@clk9
63   always@clk10
64   always@clk11
65   always@clk12
66   always@clk13
67   always@clk14
68   always@clk15
69   always@clk16
70   always@clk17
71   always@clk18
72   always@clk19
73   always@clk20
74   always@clk21
75   always@clk22
76   always@clk23
77   always@clk24
78   always@clk25
79   always@clk26
80   always@clk27
81   always@clk28
82   always@clk29
83   always@clk30
84   always@clk31
85   always@clk32
86   always@clk33
87   always@clk34
88   always@clk35
89   always@clk36
90   always@clk37
91   always@clk38
92   always@clk39
93   always@clk40
94   always@clk41
95   always@clk42
96   always@clk43
97   always@clk44
98   always@clk45
99   always@clk46
100  always@clk47
101  always@clk48
102  always@clk49
103  always@clk50
104  always@clk51
105  always@clk52
106  always@clk53
107  always@clk54
108  always@clk55
109  always@clk56
110  always@clk57
111  always@clk58
112  always@clk59
113  always@clk60
114  always@clk61
115  always@clk62
116  always@clk63
117  always@clk64
118  always@clk65
119  always@clk66
120  always@clk67
121  always@clk68
122  always@clk69
123  always@clk70
124  always@clk71
125  always@clk72
126  always@clk73
127  always@clk74
128  always@clk75
129  always@clk76
130  always@clk77
131  always@clk78
132  always@clk79
133  always@clk80
134  always@clk81
135  always@clk82
136  always@clk83
137  always@clk84
138  always@clk85
139  always@clk86
140  always@clk87
141  always@clk88
142  always@clk89
143  always@clk90
144  always@clk91
145  always@clk92
146  always@clk93
147  always@clk94
148  always@clk95
149  always@clk96
150  always@clk97
151  always@clk98
152  always@clk99
153  always@clk100
154  always@clk101
155  always@clk102
156  always@clk103
157  always@clk104
158  always@clk105
159  always@clk106
160  always@clk107
161  always@clk108
162  always@clk109
163  always@clk110
164  always@clk111
165  always@clk112
166  always@clk113
167  always@clk114
168  always@clk115
169  always@clk116
170  always@clk117
171  always@clk118
172  always@clk119
173  always@clk120
174  always@clk121
175  always@clk122
176  always@clk123
177  always@clk124
178  always@clk125
179  always@clk126
180  always@clk127
181  always@clk128
182  always@clk129
183  always@clk130
184  always@clk131
185  always@clk132
186  always@clk133
187  always@clk134
188  always@clk135
189  always@clk136
190  always@clk137
191  always@clk138
192  always@clk139
193  always@clk140
194  always@clk141
195  always@clk142
196  always@clk143
197  always@clk144
198  always@clk145
199  always@clk146
200  always@clk147
201  always@clk148
202  always@clk149
203  always@clk150
204  always@clk151
205  always@clk152
206  always@clk153
207  always@clk154
208  always@clk155
209  always@clk156
210  always@clk157
211  always@clk158
212  always@clk159
213  always@clk160
214  always@clk161
215  always@clk162
216  always@clk163
217  always@clk164
218  always@clk165
219  always@clk166
220  always@clk167
221  always@clk168
222  always@clk169
223  always@clk170
224  always@clk171
225  always@clk172
226  always@clk173
227  always@clk174
228  always@clk175
229  always@clk176
230  always@clk177
231  always@clk178
232  always@clk179
233  always@clk180
234  always@clk181
235  always@clk182
236  always@clk183
237  always@clk184
238  always@clk185
239  always@clk186
240  always@clk187
241  always@clk188
242  always@clk189
243  always@clk190
244  always@clk191
245  always@clk192
246  always@clk193
247  always@clk194
248  always@clk195
249  always@clk196
250  always@clk197
251  always@clk198
252  always@clk199
253  always@clk200
254  always@clk201
255  always@clk202
256  always@clk203
257  always@clk204
258  always@clk205
259  always@clk206
260  always@clk207
261  always@clk208
262  always@clk209
263  always@clk210
264  always@clk211
265  always@clk212
266  always@clk213
267  always@clk214
268  always@clk215
269  always@clk216
270  always@clk217
271  always@clk218
272  always@clk219
273  always@clk220
274  always@clk221
275  always@clk222
276  always@clk223
277  always@clk224
278  always@clk225
279  always@clk226
280  always@clk227
281  always@clk228
282  always@clk229
283  always@clk230
284  always@clk231
285  always@clk232
286  always@clk233
287  always@clk234
288  always@clk235
289  always@clk236
290  always@clk237
291  always@clk238
292  always@clk239
293  always@clk240
294  always@clk241
295  always@clk242
296  always@clk243
297  always@clk244
298  always@clk245
299  always@clk246
300  always@clk247
301  always@clk248
302  always@clk249
303  always@clk250
304  always@clk251
305  always@clk252
306  always@clk253
307  always@clk254
308  always@clk255
309  always@clk256
310  always@clk257
311  always@clk258
312  always@clk259
313  always@clk260
314  always@clk261
315  always@clk262
316  always@clk263
317  always@clk264
318  always@clk265
319  always@clk266
320  always@clk267
321  always@clk268
322  always@clk269
323  always@clk270
324  always@clk271
325  always@clk272
326  always@clk273
327  always@clk274
328  always@clk275
329  always@clk276
330  always@clk277
331  always@clk278
332  always@clk279
333  always@clk280
334  always@clk281
335  always@clk282
336  always@clk283
337  always@clk284
338  always@clk285
339  always@clk286
340  always@clk287
341  always@clk288
342  always@clk289
343  always@clk290
344  always@clk291
345  always@clk292
346  always@clk293
347  always@clk294
348  always@clk295
349  always@clk296
350  always@clk297
351  always@clk298
352  always@clk299
353  always@clk300
354  always@clk301
355  always@clk302
356  always@clk303
357  always@clk304
358  always@clk305
359  always@clk306
360  always@clk307
361  always@clk308
362  always@clk309
363  always@clk310
364  always@clk311
365  always@clk312
366  always@clk313
367  always@clk314
368  always@clk315
369  always@clk316
370  always@clk317
371  always@clk318
372  always@clk319
373  always@clk320
374  always@clk321
375  always@clk322
376  always@clk323
377  always@clk324
378  always@clk325
379  always@clk326
380  always@clk327
381  always@clk328
382  always@clk329
383  always@clk330
384  always@clk331
385  always@clk332
386  always@clk333
387  always@clk334
388  always@clk335
389  always@clk336
390  always@clk337
391  always@clk338
392  always@clk339
393  always@clk340
394  always@clk341
395  always@clk342
396  always@clk343
397  always@clk344
398  always@clk345
399  always@clk346
400  always@clk347
401  always@clk348
402  always@clk349
403 
```

```

58      end
59
60      always@(posedge clk)
61      begin if (opt == 000000) begin ask1 <= ~ask0 and cnot0 and else cnot <= cnot1 and
62
63      always*
64      begin
65          case(cnot)
66              0'b0000 : ask1 <= ~b01101111;
67              0'b0001 : ask1 <= ~b00001111;
68              0'b0010 : ask1 <= ~b01101111;
69              0'b0011 : ask1 <= ~b01001111;
70              0'b0100 : ask1 <= ~b11001111;
71              0'b0101 : ask1 <= ~b11101111;
72              0'b0110 : ask1 <= ~b11111111;
73              0'b0111 : ask1 <= ~b11111111;
74              0'b1000 : ask1 <= ~b11111111;
75              0'b1001 : ask1 <= ~b11101111;
76              0'b1010 : ask1 <= ~b11001111;
77              default : ask1 <= ~b00000000;
78          endcase
79      end
80
81      always*
82      begin
83          case(ask)
84              0'b0000 : ask2 <= ~b01101111;
85              0'b0001 : ask2 <= ~b00001111;
86              0'b0010 : ask2 <= ~b01101111;
87              0'b0011 : ask2 <= ~b01001111;
88              0'b0100 : ask2 <= ~b11001111;
89              0'b0101 : ask2 <= ~b11101111;
90              0'b0110 : ask2 <= ~b11111111;
91              0'b0111 : ask2 <= ~b11111111;
92              0'b1000 : ask2 <= ~b11111111;
93              0'b1001 : ask2 <= ~b11101111;
94              0'b1010 : ask2 <= ~b11001111;
95              default : ask2 <= ~b00000000;
96          endcase
97      end
98
99

```

สรุปผลการทดลอง

จากโค้ดโปรแกรมในการทดลองนั้นเราได้แบ่งการทำงานเป็นขั้นตอนโดยโค้ดชุดแรกนั้นเป็นการทำงานให้บอร์ด FPGA นั้นนับเลข 0 – 9 เพื่อให้ทำงานได้ตามปกติโดยใช้สัญญาณนาฬิกา มาช่วยหลังจากนั้นได้ทำการเพิ่มหลักสิบเพิ่มเพื่อให้นับ 0 – 99 โดยเราได้ทำเป็น state ทั้งหมด 2 state โดยจากโค้ดนั้นเราได้มีเงื่อนไขต่างๆในการเช็ค ดังนี้

1. ถ้าหลักหน่วยเป็น 9 จะให้เพิ่มหลัก 10 อีก 1 ค่า
2. ถ้าทั้งสองหลักเป็นเลข 9 ทั้งคู่ให้กลับมากลายเป็นเลข 0

จากเงื่อนไขหลังจากนั้นจะทำการเข้า state ทั้งสอง state ดังนี้

1. ถ้าหลักหน่วยทำงานครบแล้วให้เพิ่มหลักสิบ 1 ค่า
2. หลังจากทำงานที่หลักสิบให้กลับไปทำงานที่หลักหน่วย

ทั้งสอง state นี้ทำงานร่วมกันสลับทำไปเรื่อยๆจนถึง 99 และหลังจากนั้นจะมีเสียงของ buzzer ดังขึ้น และดังต่อไปเรื่อยๆนั่นเอง

